

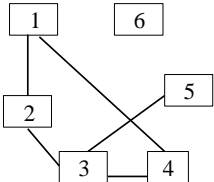
Grafuri neorientate

Graf neorientat = un graf $G=(X, U)$ în care fiecare muchie este simetrică: $(x,y) \in U$ atunci $(y,x) \in U$.

Grad = Gradul unui nod v , dintr-un graf neorientat, este un număr natural ce reprezintă numărul de noduri adiacente cu acesta (sau numărul de muchii incidente cu nodul respectiv)

Nod izolat = Un nod cu gradul 0.

Nod terminal = un nod cu gradul 1

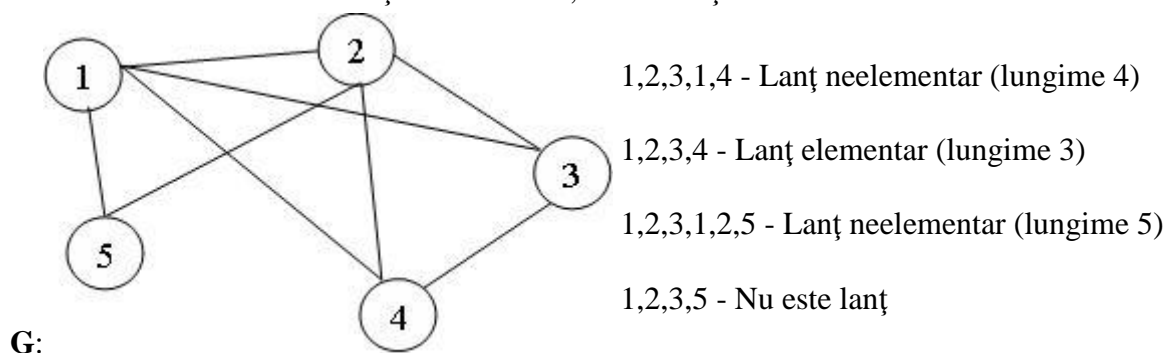
	<p>Nodul 5 este terminal (gradul 1).</p> <p>Nodul 6 este izolat (gradul 0)</p> <p>Nodurile 1, 2, 4 au gradele egale cu 2.</p>
---	---

Lanț = Se numește lanț o succesiune de noduri $x_1 \dots x_k$, cu proprietatea că oricare două noduri vecine (x_i, x_{i+1}) aparțin de B .

- x_1, x_k sunt extremitățile lanțului.

Lungimea lanțului este egală cu numărul de muchii care îl compun, $k-1$.

Dacă nodurile din lanț sunt distincte, atunci lanțul este **elementar**.

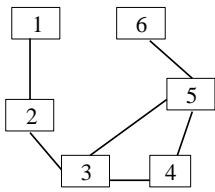


Lanț simplu = lanțul care conține numai muchii distincte

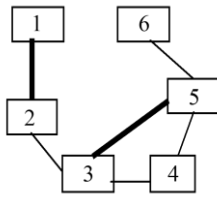
Lanț compus = lanțul care nu este format numai din muchii distincte

Ciclu = Un lanț în care primul nod coincide cu ultimul.

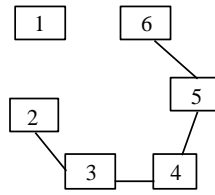
Ciclul este **elementar** dacă este format doar din noduri distincte, excepție făcând primul și ultimul. Lungimea unui ciclu **nu poate fi 2**.

	<p>Succesiunea de vârfuri 2, 3, 5, 6 reprezintă un lanț simplu și elementar de lungime 3.</p> <p>Lanțul 5 3 4 5 6 este simplu dar nu este elementar.</p> <p>Lanțul 5 3 4 5 3 2 este compus și nu este elementar.</p> <p>Lanțul 3 4 5 3 reprezintă un ciclu elementar</p>
---	--

Graf parțial = Dacă dintr-un graf $G=(X,U)$ se suprimă cel puțin o muchie atunci noul graf $G'=(X,U')$, $U' \subset U$ se numește **graf parțial** al lui G .

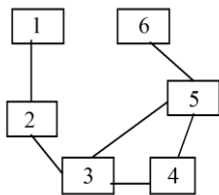


G

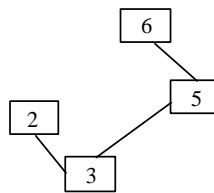


$G1$ este graf parțial al lui G

Subgraf = Dacă dintr-un graf $G=(X,U)$ se suprimă cel puțin un nod împreună cu muchiile incidente lui, atunci noul graf $G'=(X',U')$, $X' \subset X$ și $U' \subset U$ se numește **subgraf** al lui G .

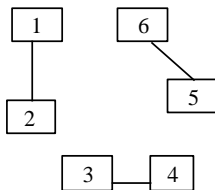


G

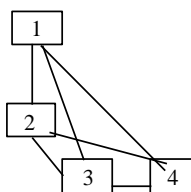


$G1$ este subgraf al lui G

Graf regulat = graf neorientat în care toate nodurile au același grad;



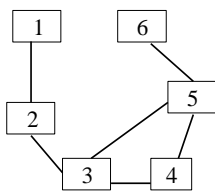
Graf complet = graf neorientat $G=(V,E)$ în care există muchie între oricare două noduri. Numărul de muchii ale unui graf complet este: $n*(n-1)/2$, unde n este numărul de noduri



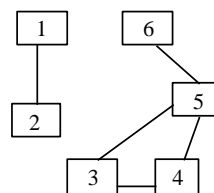
graf complet. Nr de muchii: $m = 4 \times (4-1) / 2 = 6$

Graful complet cu n vârfuri se notează cu K_n .

Graf conex = graf neorientat $G=(X,U)$ în care pentru orice pereche de noduri (x,y) există un lanț care le unește.

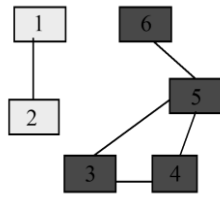


graf conex



nu este graf conex

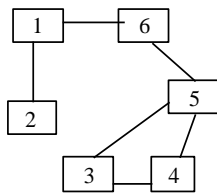
Componentă conexă = subgraf al grafului de referință, maximal în raport cu proprietatea de conexitate (între oricare două vârfuri există lanț);



graful nu este conex. Are 2 componente conexe:
1, 2 și 3, 4, 5, 6

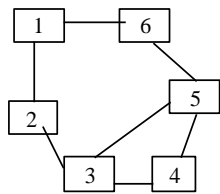
Graf hamiltonian

Lanț hamiltonian = un lanț elementar care conține toate nodurile unui graf



$L=[2, 1, 6, 5, 4, 3]$ este lanț hamiltonian

Ciclu hamiltonian = un ciclu elementar care conține toate nodurile grafului

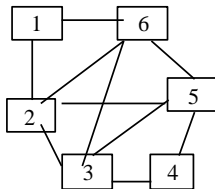


$C=[1,2,3,4,5,6,1]$ este ciclu hamiltonian

Graf hamiltonian = un graf G care conține un ciclu hamiltonian
Graful anterior este graf hamiltonian.

Graf eulerian

Lanț eulerian = un lanț simplu care conține toate muchiile unui graf



Lantul: $L=[1.2.3.4.5.3.6.2.5.6]$ este lant eulerian

Ciclu eulerian = un ciclu simplu care conține toate muchiile grafului
Ciclul: $C=[1.2.3.4.5.3.6.2.5.6.1]$ este ciclu eulerian

Graf eulerian = un graf care conține un ciclu eulerian.

Condiție necesară și suficientă:

Un graf este **eulerian** dacă și numai dacă oricare vârf al său are **gradul par**.

Parcurgerea grafurilor neorientate

Explorarea în lățime a grafurilor – Algoritmul BF (Breadth First)

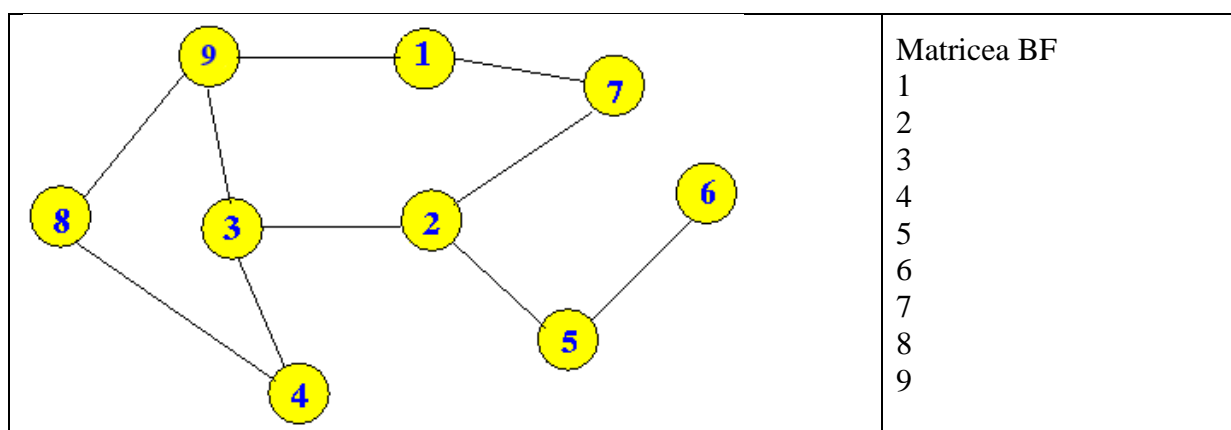
La explorarea în lățime, după vizitarea nodului inițial, se explorează toate nodurile adiacente lui, se trece apoi la primul nod adiacent nevizitat și se explorează toate nodurile adiacente acestuia și neparcurse încă, ș.a.m.d.

Algoritmul

Se va folosi o coadă memorată într-un vector în care se scriu nodurile în ordinea în care sunt parcurse: nodul inițial v (de la care se pornește), apoi nodurile a, b, \dots , adiacente lui v , apoi cele adiacente lui a , cele adiacente lui b, \dots , ș.a.m.d.

Coadă este folosită astfel:

- se pune primul nod în coadă;
- se află toate vârfurile adiacente cu primul nod din coadă și se adaugă la sfârșitul cozii cele nevizitate încă
- se trece la următorul nod și i se află nodurile adiacente
- procesul se repetă până când se ajunge la sfârșitul cozii



Explorarea în adâncime a grafurilor – Algoritmul DF (Depth First)

Parcurgerea unui graf în adâncime se face prin utilizarea stivei, alocate explicit sau implicit prin subprograme recursive).

Pentru fiecare vârf se parcurge primul dintre vecinii lui neparcurși încă. După vizitarea vârfului inițial x_1 , se explorează primul vârf adiacent lui, fie acesta x_2 , se trece apoi la primul vârf adiacent cu x_2 și care nu a fost parcurs încă, ș.a.m.d.

Pentru a nu trece de două ori prin același vârf se va folosi un vector s , cu valoarea:

$s[k] = 0$, dacă vârful k nu a fost explorat

$s[k] = 1$, dacă vârful k a fost explorat

